IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Appling		
Serial No.:	09/747,366	Art Unit:	2173
Filed:	December 22, 2000	Examiner:	Bonshock, Dennis G.

For: UPDATING OBJECTS CONTAINED WITHIN A WEBPAGE

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

DECLARATION UNDER 37 C.F.R. § 1.131

Sir:

I, Stephen Charles Appling, hereby declare that:

1. I am the sole inventor of U.S. Patent Application No. 09/747,366 ("the '366 application") referenced above.

2. I have reviewed U.S. Patent No. 6,823,359 to Heidingsfeld et al. ("Heidingsfeld"), which was cited in the Office Actions mailed June 2, 2005 and December 23, 2005, in connection with the above referenced patent application. I understand that Heidingsfeld has an effective prior art date of November 21, 2000.

3. The attached documents demonstrate that I conceived of the system and method for updating objects contained within a webpage defined by pending Claims 1-5, 7-11, 13, and 16-21 of the '366 application prior to November 21, 2000. The documents are described below.

4. Exhibit A is a facsimile sent on September 8, 2000 from counsel addressed to me that includes a copy of an earlier draft of the '366 application. The earlier draft is titled "A METHOD AND SYSTEM FOR UPDATING OBJECTS CONTAINED WITHIN A WEB

PAGE" ("Application Draft"), and describes the system and method for selectively updating objects on a web page automatically without the need for refreshing the entire web page. The figures included with this Application Draft are substantially similar to the figures submitted with the '366 application filed on December 22, 2000. Moreover, the contents of the Application Draft, including the accompanying figures, provide basis for the claimed inventions in Claims 1-21 of the '366 application.

5. While the Application Draft of Exhibit A evidences my conception of the claimed inventions in Claims 1-21 of the '366 application prior to the filing date of Heidingsfeld, the Application Draft should not be considered an admission that September 8, 2000 is the earliest date of my conception of the claimed inventions in Claims 1-21 of the '366 application.

6. Exhibit B contains correspondences dated November 10, 2000, December 11, 2000, and December 22, 2000 that are communications with the patent attorneys assisting in the drafting of the '366 application regarding revisions to the '366 application.

7. To preserve the confidentiality of communications with counsel and prevent waiver of the attorney-client privilege, portions of the correspondences of Exhibit B have been redacted to provide no more information than is necessary to establish diligence from a date prior to the filing of Heidingsfeld on November 21, 2000 up to the filing of the '366 application on December 22, 2000.

8. At the time of the correspondence dated November 10, 2000, I was an employee of Automated Logic Corporation. The correspondence dated November 10, 2000 is between counsel assisting with the preparation of the '366 application and Eric Craton. At the time of the correspondence dated November 10, 2000, Mr. Craton was a vice president of Automated Logic

Corporation assisting me in the preparation of the '366 application. Mr. Craton is also carbon copied on the email correspondence of December 11, 2000 and one of the correspondences of December 22, 2000.

9. At a date between the correspondence dated November 10, 2000 and the correspondence dated December 11, 2000, the law firm handling the preparation of the '366 application was changed from Kilpatrick Stockton LLP to Sutherland Asbill & Brennan LLP. As a result, the November 10, 2000 correspondence was with a different attorney than the December 11, 2000 and December 22, 2000 correspondences.

10. The '366 application submitted to the USPTO on December 22, 2000 is referred to as the "IFRAME" application in the email correspondences of December 11, 2000 and December 22, 2000.

11. The correspondences of November 10, 2000, December 11, 2000, and December 22, 2000 corroborate that I, along with the assistance of counsel, did diligently pursue the drafting and revising of the '366 application from at least a date just prior to November 21, 2000 until the actual filing of the '366 application on December 22, 2000.

12. The research and development of the invention described in the Exhibit documents was conducted in the United States after January 1, 1996.

13. I declare that all statements made herein of my own knowledge and belief are true and that all statements made on information and belief are believed to be true, and further that the statements are made with the knowledge that willful false statements are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code, and that such

willful false statements may jeopardize the validity of the application or any patent issuing thereon.

Stephen Charles Appling

3/10/06

Stephen Charles Appling | Date

EXHIBIT A

* * * COMMUNICATION RESULT REPORT (SEP. 8. 2000 6:10PM) * * *

LE MODE	OPTION	ADDRESS (GROUP)	TTI RESULT	PAGE
74 MEMORY TX		*601360011097707953580	OK	27/27

REASON FOR ERROR
E-1) HANG UP OR LINE FAIL
E-2) BUSY
E-3) NO ANSWER
E-4) NO FACSIMILE CONNECTION

JONES & ASKEW
LIMITED LIABILITY PARTNERSHIP

2400 Monarch Tower
3424 Peachtree Road, N.E.
Atlanta, Georgia 30326

Tel 404-949-2400
Fax 404-949-2499

www.jonesaskew.com

Date: September 8, 2000 Fax No: 770-795-3580

To: Steve Appling

From: S. Craig Hemenway Our Ref: 01360-0110

Number of Pages Transmitted: 27

If you do not receive all pages, please contact us immediately.

MESSAGE:

JONES & ASKEW
LIMITED LIABILITY PARTNERSHIP

2400 Monarch Tower
3424 Peachtree Road, N.E.
Atlanta, Georgia 30326

Tel 404-949-2400
Fax 404-949-2499

www.jonesaskew.com

Date: September 8, 2000 Fax No: 770-795-3580

To: Steve Appling

From: S. Craig Hemenway Our Ref: 01360-0110

Number of Pages Transmitted: 27

If you do not receive all pages, please contact us immediately.

MESSAGE:

A METHOD AND SYSTEM FOR UPDATING OBJECTS CONTAINED WITHIN A WEB PAGE

TECHNICAL FIELD

This application relates generally to a method and system for selectively updating objects on a web page, and more specifically to providing automatic updating of web page objects without the need for refreshing the entire web page.

BACKGROUND

Oftentimes, the content of a web page is dynamic in nature. In some cases, changes to the content occur every few moments, as updated information is placed on a web page. For example, stock market tickers and live camera feeds seen across the Internet may constantly send updated information to a web browser. This permits a continuous media stream, which in turn keeps an observer informed of up-to-the-minute or time-critical events.

Many current implementations of dynamic web pages require that the web page be frequently refreshed in order to display the newest

information. This typically requires that the user continually click a "refresh" button, consuming the user's time and attention. Additionally, this implementation requires that the entire web page be reloaded, introducing significant time losses from retransmitting information that has not changed. A web page employing a timer contained within the page to periodically request updated information from a server also suffers from the drawback of reloading the entire web page.

Alternate methods of continuous updating include the use of Java applets to receive information broadcast by an application running on a remote server, hereinafter referred to as a "server-side application." Typically, the server-side application generates dynamically updated content which is then requested by or transmitted to a Java applet executed within a web browser. By implementing a server-side application, the requirement of user intervention to refresh a page is eliminated. However, this implementation has a different set of drawbacks: namely, that the Java applet typically only makes changes to itself or objects displayed within the applet. In the case where an object requiring updating is not a part of the applet, a user-initiated manual refresh is again required, or the Java applet must make use of the Java feature known as "LiveConnect". Further, many corporate and educational firewalls do not permit Java applets to receive continuously broadcast information from a remote server.

Thus, there is a need in the art for a method of updating HTML objects contained within a web page without requiring either a manual command or the use of a Java applet.

SUMMARY OF THE INVENTION

Briefly described, the present invention may constantly update selective portions of a web page without requiring that the entire document be reloaded, and without user intervention. Generally speaking, the invention takes the form of a web page containing an inline frame, or "I-frame", displayed in a web browser. A frame is a method of displaying data within a web page well known to those skilled in the art. An exemplary embodiment of the current invention uses HTML commands to set the I-frame's height and width to zero. This I-frame, therefore, may not be seen by a user of the present invention looking at a displayed web page. The web page also typically contains one or more updateable HTML objects, and may contain one or more nonupdateable HTML objects. Note that, although the term "I-frame" is used throughout this specification, that the invention may be practiced with any type of HTML-defined frame. Additionally, alternate embodiments of the present invention may make use of frames with dimensions other than a height and width of zero, without affecting the performance of the invention. Thus the term "I-frame" is used for consistency and convenience only.

The code defining an I-frame may contain a timer. In this event, the timer runs until a predetermined value is met, at which point the I-frame requests updated data from a server. Typically, this data takes the form of instructions to the I-frame to alter the various updateable HTML objects contained within the web page but external to the I-frame. Thus, the I-frame may alter portions of the web page which it are contained within it, and with which it otherwise has no direct interaction.

That the invention improves over the drawbacks of prior web page update systems and accomplishes the advantages described above

will become apparent from the following detailed description of the embodiments and the appended drawings and claims.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 displays an exemplary operating environment of the current invention.

FIG. 2 displays an HTML frame incorporating an exemplary embodiment of the current invention.

FIG. 3 displays an initial HTML frame in accordance with an exemplary embodiment of the current invention.

FIG. 4 displays an updated HTML frame in accordance with an exemplary embodiment of the current invention.

FIG. 5 is a flowchart depicting the operation of an exemplary embodiment of the current invention.

DETAILED DESCRIPTION OF THE EMBODIMENT

The present invention is directed to a method and system for providing content updates to a web page displayed in a web browser, without the need for refreshing the entire web page or requiring any user input.

The exemplary embodiment of the present invention takes the form of a web page including selectively refreshed display objects and an inline frame, or "I-frame". An I-frame is a variant of a normal frame. Within the context of a web page, a frame is a portion of the web browser dedicated to displaying a first (or child) web page, such as an HTML page, within a second (or parent) web page. Thus, each frame may contain a separate web page, sharing display space within the web browser with other frames. Operations within a first frame may affect

the display in a second frame, although generally only through user input. For example, a first frame may display a table of contents, while the second frame displays individual chapters. The chapter displayed in the second frame may be updated by means of the user selecting a specific portion of the table of contents.

In an exemplary embodiment, the I-frame has both a height and width of zero, thus rendering it invisible within the browser. The exemplary embodiment further creates the I-frame as an HTML construct.

In conjunction with the web browser, the I-frame is capable of retrieving information and/or instructions from a remote server hosting a web page, Java servlet, File Transfer Protocol ("FTP") data structure, or other shared data service. Typically the I-frame is operative to contact a server at fixed intervals, at which point the server passes data (possibly generated by a Java servlet) to the I-frame. In an exemplary embodiment, the I-frame contacts the server using the hypertext transfer protocol ("HTTP") or other protocols well known to those skilled in the art. Alternate embodiments may use different data transfer protocols to effect the I-frame request and server response.

The updated data transmitted by the server typically includes instructions which permit the I-frame to modify various elements of the parent web page that contains the I-frame. In this manner, selective portions of the web page may be updated without requiring a complete refresh of the entire HTML document or any user intervention.

Referring now to the drawings, in which like numerals represent like elements throughout the several figures, aspects of the present invention and a suitable operating environment will be described.

Exemplary Operating Environment

FIG. 1 and the following discussion are intended to provide a general description of the suitable computing environment in which the invention may be implemented. While the invention will be described in the general context of a web page displayed within an application program that runs on an operating system in conjunction with a personal computer, the invention may also be implemented in additional manners well known to the invention. Moreover, those skilled in the art will appreciate that the invention may be practiced with other computer system configurations, minicomputers, mainframe computers, and the like. The invention may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communication network.

The present invention may be embodied as a web page object update system **100**. The web page object update system **100** comprises a web page **230** and a server-side application **102**. The server-side application **102** resides on a server **104**, and is typically created using the Java programming language. Alternate embodiments may employ different types of server-side applications, which may be created using different programming languages. For example, server-side applications may consist of Java scripts (or "servlets"), CGI scripts, or any script capable of generating commands interpretable and executable by the web page object update system **100**.

The server **104** further typically contains one or more web page data files **103** which, when loaded into a web browser **112**, may implement the web page **230** comprising a portion of the web page object update system **100**. The website **114** and constituent web page data files **103** may take multiple forms, such as an HTML instruction

set, an XML instruction set, a server hypertext markup language ("SHTML") instruction set, and so on. Accordingly, it should be understood that the spirit of this invention embraces any web page object update system **100** created through the use of any hypertext-like language, such as those mentioned above. Alternate embodiments may store the web page data files **103** locally, on another server, or separated into multiple files resident on different elements of a network **106**, described in greater detail below.

The server **104** is in communication with a network **106**, such as the Internet, an intranet, a telephone wireline and/or wireless network, a cellular network, a broadband network, and any other wireless or wire-based network. The clients **108a - 108n** are also in communication with the network. Clients **108a - 108n** may access the web page data file **103** by using a web browser **110**, or similar application program, to connect to a website **114** linking the server **104** and the distributed computer network **106**. Once the website **114** is accessed by the web browser **112**, a web page **230** is created within the browser by rendering the web page data file **103**. Typically, a user may view the web page **230** through a display monitor **116** connected to the client **108a**. An example of the web page **230** shown on the display monitor **116** is shown in FIG. 2.

Description of an Exemplary Embodiment

FIG. 2 displays an exemplary embodiment of the present invention, embodied as a web page **230** displayed within a web browser **112**. Although the exemplary embodiment makes use of HTML, alternate embodiments may be coded in XML, SHTML, DHTML, Lynx, or any other hypertext-like language. Thus, the term "web page" is used

throughout this specification to embrace all forms of web pages programmed in any language.

The web page **230** contains an I-frame **200**, a non-updateable object **220**, and an updateable object **210**. The non-updateable object **220** signifies a portion of the web page **230** displaying content that will not be altered through the operation of the I-frame **200**. By contrast, the I-frame **200** may and typically will change the updateable object **210** at some point during operation of the web object update system **100**. Note that the dashed lines surrounding the I-frame **200**, updateable object **210,** and the non-updateable object **220** are for illustrative purposes only, and need not be displayed in an embodiment of the present invention.

Upon initial display of the web page **230** by the web browser **112**, the web object update system **100** sets a timer (not shown), contained within the code that defines the I-frame **200**, to zero and begins a count. When the timer reaches a threshold value specified in the code comprising the I-frame **200**, the I-frame **200** requests updated data from a remote server **104**. Typically, this updated data is generated by the server-side application **102** stored on the remote server **104**. The updated data may take the form of JavaScript generated by a Java servlet. Regardless of the format, the updated data carries instructions to the I-frame **200** which order the I-frame **200** to modify certain objects of the parent web page **230**. The updated data may alternately take the form of raw data relayed from a remote network element, HTML code, XML code, and so on. In an alternate embodiment, the I-frame **200** may be self-determinative in that the I-frame may be configured to process the raw data to determine which portions of the parent web page **230** to update and to generate instructions accordingly; this alternative

operation is to be distinguished from receiving instructions from the server **104**.

In an exemplary embodiment, the timer is maintained in the code defining the I-frame, although alternate embodiments may create the timer within the code defining the parent web page **230**. In an exemplary embodiment, the timer is located at the client **108** and not the server **104**, in order to avoid implementing a "server push" scheme. "Server push" refers to the transmission of data from a server **104** either continuously or at periodic intervals, without any data request from the web browser **112**. An exemplary embodiment avoids a server push due to the fact that server pushes are blocked by certain firewalls. Further, some ISPs and/or clients **108a** may time out and disconnect from the network **106** unless the web browser **112** occasionally transmits a data request to a server **104**. By avoiding the server push implementation, the possibility of client timeout is minimized.

The exemplary implementation shown in FIG. 2 is a simple version of the web object update system **100** shown in Fig.1. The I-frame **200** may receive updated data from a remote server **104** at set intervals, which enables the I-frame **220** to update a display of the current time in the parent web page **230**. The current time is an example of an updateable object **210**, while the phrase "The time is:" is an example of a non-updateable object **220**. Thus, whenever the timer reaches a threshold value, the I-frame **200** requests updated data, and accordingly updates the time displayed an updateable object **210** without requiring the user to refresh the entire web page **230**.

The HTML source code necessary to create the exemplary web page **230** and I-frame **200** of Fig. 2 is as follows:

```
<HTML>
```

```
<BODY>
  <IFRAME src="time.jsp" width="0" height="0"
style="visibility:hidden"></IFRAME>
  The time is : <SPAN id="timeDiv">???</SPAN>
</BODY>
</HTML>
```

The Java ServerPage ("JSP") application **102** generates instructions periodically received by the I-frame **200** is as follows:

```
<HTML>
<SCRIPT>
     window.parent.timeDiv.innerHTML = "<%= new
java.util.Date() %>";
     function refresh() { location.href =
location.href; }
     setTimeout(refresh, 2000);
</SCRIPT>
</HTML>
```

Again, alternate embodiments may use different forms of server-side applications **102** without departing from either the spirit or scope of the invention.

The web page object update system **100** may be used to monitor and update more than a single updateable object **210** of a web page **230**. For example, one possible application of the system is in the monitoring and display of airflow within a building, as shown in FIG. 3. In the example of FIG. 3, a browser **112** displays a web page **230** including a HVAC schematic **300** that represents the layout pipes within an office building. It is to be understood that the representation is not limited to just an office building, but could also be a home, warehouse, or any other building or facility. Each data point, such as the incoming airflow **310** and airflow temperature **320**, within the schematic **300** may be represented by an updateable individual HTML object, which may be changed in some manner as actual conditions within the HVAC system

change. For example, the air temperature and airflow readouts may change with as airflow within the pipe changes.

The building may be collocated at the client **108**, the remote server **104**, or may be in another location. Regardless of the building location, various portions of a monitored section of an HVAC system may contain a small sensor which monitors temperature, humidity, and/or other factors important in maintaining properly balanced airflow. It is well known that such sensors may periodically relay monitored data to the server **104** to record changes in comfort or environmental control levels. The server **104** may then employ this monitored data to generate updated data for display on the web page **230**. In turn, the I-frame **200** periodically requests the updated data from the server **104**, along with instructions regarding which portions of the schematic **300** require updating to show changed conditions. As necessary, the I-frame **200** updates each individual updateable object **210** to reflect changes in comfort level.

In FIG. 3, the initial state of an exemplary embodiment of an HTML content update system **100** is displayed as a web page **230** in a web browser **112**. For purposes of this example, presume that web page **230** displays a HVAC pipe schematic **300**, corresponding to an actual pipe section located within a building. The web page displays the incoming airflow **310**, the airflow temperature **320**, the main outgoing airflow **330**, and the sectional outgoing airflow **340**. The web page may further include several updateable objects **210**. Sample updateable objects include the incoming airflow **310**, the airflow temperature **320**, the main outgoing airflow **330**, the sectional outgoing airflow **340**, the valve position **350**, and the valve display **360**. For example, in FIG. 3 the incoming airflow is 2300 cubic feet per minute (CFM), the airflow

temperature is 62 degrees Fahrenheit, and the valve position is 43% open. The valve display **360** may be animated in the exemplary embodiment to show the valve opening and closing in a real-time fashion.

When the web page **230** is first loaded, a timer within the I-frame **200** begins counting. Such timers are well known to those of ordinary skill in the art. Once the timer reaches a configurable threshold value, the I-frame **200** sends a request for updated data to the server **104**. The server **104** relays to the I-frame **200** instructions to modify the various updateable objects within the schematic **300**. For example, the I-frame **200** may receive instructions from the server to change the incoming airflow **310**, because the amount of air passing through the pipe has changed. Alternate embodiments may not relay updated data, but rather raw (or processed) monitored data which may then be used by the I-frame **200** itself to determine the proper method of updating. That is, in an exemplary embodiment decisions as to what updateable objects **210** are updated, and thus what instructions are generated to cause such updates, are made by the server-side application **102** resident on the server **104**. The instructions passed to the I-frame **200** are a result of this decision process. Alternate embodiments may pass data other than instructions from the servlet **102** to the I-frame **200**, and permit the client **108a** to determine what updateable objects **210**, if any, must be changed.

FIG. 4 shows the browser **112** displaying an updated web page **230** comprising an updated HVAC schematic **300′**, which represents the schematic **300** of FIG. 3, after certain changes have taken place. In the example shown in FIG. 4, presume two conditions have changed. First, the airflow temperature **320′** has risen by five degrees Fahrenheit.

Second, the valve **360′** has been partially closed. In a typical embodiment, remote sensors located within the HVAC pipe may detect these changes and relayed updated data to the server **104**, which in turn may generate updated data to be sent to the I-frame **200**.

In modifying the web page **230** to reflect these changes, the I-frame **200** has updated various updateable objects **210**. In this case, the I-frame **200** has received instructions to modify the airflow temperature **320′**, the main outgoing airflow **330′**, the sectional outgoing airflow **340′**, the valve position **350′**, and the valve display **360′**. The partial closing of the valve diminishes the airflow in the sectional portion of the HVAC pipe, and accordingly increases the main outgoing airflow. Once the I-frame **200** executes its timed request to the server **104**, the new instruction set is passed to the I-frame **200**, which in turn makes the necessary changes to the updateable objects **320′**, **330′**, **340′**, **350′**, and **360′**.

In FIG. 4, the web page object update system **100** has altered the various values in response to the received instruction set. Specifically, the valve display **360′** has been changed to show that the valve has partially closed, and all relevant temperature and airflow readings are altered to reflect current conditions. In this manner, a user may have accurate information regarding the current airflow in a remote building displayed in an easy to recognize format, without manually requesting new data and without having to refresh the web page **230**.

Although the above example includes only temperature, position, and airflow as variables, many other factors may be monitored and updated by the web page object update system **100**. For example, humidity, chemical composition, volume, ambient noise level, and

luminosity all may be monitored factors in various systems compatible with the web page object update system.

FIG. 5 shows an exemplary method **500** detailing the logical operation of the web page object update system **100**. The method **500** begins with step **501**, in which a web page **230** is loaded into a web browser **112** and displayed on a display monitor **116**. Once the web page is loaded and displayed, method **500** advances to step **505**, where an I-frame **200** is created within the web page **230**. Next, step **510** is executed, wherein a timer is initiated. This timer continues to operate until a threshold value is reached. This threshold value is configurable by an administration of the system. Following step **510**, the method proceeds to decision step **520**, where it is determined whether the threshold value of the timer has been reached. If the threshold value has not been reached, then step **520** is repeated such that the timer continues unabated until it reaches the threshold value.

When the threshold value of the timer has been reached, step **525** is executed. In step **525**, the I-frame **200** initiates a "call for content update", which may comprise a communication to the server **104** for updated data. In response to the request for updated data, the client **104** receives from the server **104** the updated data and a corresponding instruction set at step **530**. As previously mentioned, the instruction set and updated data may be transmitted to the I-frame **200** from a server-side application **102**, and may comprise data instructing the I-frame to update or otherwise modify portions of the web page **230** external to the I-frame.

Alternate embodiments may pass raw (or processed) data from the server **104** to the I-frame **200**, and permit the I-frame **200** itself to interpret the data and update objects as necessary.

Once the updated data and instruction set is received, step **535** is executed. In this step, objects within the parent web page are updated, as mandated by the instruction set. Following step **535**, the method **500** returns to step **510**, where the timer is reset and the web page object update loop is repeated until the system is deactivated.

Conclusion

The web page object update system **100** may include additional functionality. For example, the timer may be activated only in response to a user input, and then deactivated in response to a second user input. Additionally, the web page object update system **100** may be used with any application capable of processing and displaying a web page, such as a word processor with appropriate extensions or an HTML editor.

Many other modifications and additional features will become evident in view of the preceding description of the embodiments of the invention. It should be understood, therefore, that the foregoing relates only to certain embodiments of the invention, and that numerous changes may be made therein without departing from the spirit and scope of the invention as defined by the following claims.

CLAIMS

We claim:

1. A method for updating objects contained within a web page, comprising:

displaying a web page;

creating a frame having a height of zero and a width of zero within the web page;

displaying outside the frame at least one updateable object within the web page;

requesting, by the frame, an instruction set from a server;

receiving the instruction set at the invisible frame; and

updating the at least one updateable object in accordance with the instruction set.

2. The method of claim 1, wherein the at least one updateable object is a HTML object.

3. The method of claim 1, further comprising the steps of:

in response to displaying the web page, starting a timer; and

in response to the timer reaching a threshold value, requesting, by the frame, an instruction set from a server.

4. The method of claim 3, further comprising, in response to requesting an instruction set from a server, resetting the timer.

5. The method of claim 1, wherein updating the at least one updateable object in accordance with the instruction set comprises updating the at least one updateable object in accordance with the instruction set without refreshing the web page.

6. A computer-readable medium containing instructions which, when executed, perform the method of claim 5.

7. The method of claim 1, wherein the frame is an HTML inline frame.

8. A method for displaying monitored factors for at least one section of a HVAC system, comprising:

monitoring at least one monitored factor in the at least one section of a HVAC system;

transmitting data representing the at least one monitored factor to a server;

within a browser in communication with the server, displaying a web page;

within the web page, creating an inline frame;

within the web page, displaying a schematic representing the at least one section of a HVAC system, the schematic having at least one updateable object representing the at least one monitored factor;

altering the at least one updateable object in order to display the current status of the monitored factor.

9. The method of claim 9, wherein the inline frame has a height of zero and a width of zero.

10. The method of claim 9, wherein the step of altering the at least one updateable object comprises in order to display the current status of the monitored factor comprises:

initiating a request, by the inline frame, for an instruction set from the server;

receiving the instruction set at the inline frame; and

updating by means of the inline frame the at least one updateable object in accordance with the instruction set.

11. The method of claim 10, further comprising:

starting a timer; and

in response to the timer reaching a threshold value, requesting, by the inline frame, the instruction set from the server.

12. The method of claim 11, further comprising, in response to requesting the instruction set from the server, resetting the timer.

13. The method of claim 8, wherein the at least one monitored factor is chosen from the group: time, temperature, flow, and position.

14. A computer-readable medium containing instructions which, when executed, perform the method of claim 12.

15. A signal set for updating at least one updateable object contained within a web page, comprising:

a first signal transmitted from a server to a client, containing instructions to the client to create a web page comprising an inline frame having a height and width of zero, and at least one updateable object; and

a second signal transmitted from the server to the client, containing an updated instruction set transmitted in response to a request for the updated instruction set from the client, the updated instruction set used by the inline frame to alter the at least one updateable object while not otherwise refreshing the web page.

16. A system for updating at least one content updateable object within a web page, comprising:

a display screen, operative to display the web page, comprising the at least one updateable object; an invisible frame having a height and width of zero;

a communications device, operative to transmit a request for updated data to a server, further operative to receive the updated data, further operative to relay the updated data to the processor;

the processor is further operative to, in response to receiving the updated data from the server, update through the invisible frame the at least one content updateable object; and

the display screen is further operative to display the at least one content updateable object without redisplaying the entirety of the web page.

17. The system of claim 16, wherein:

the processor is further operative to execute a timer in response the display screen displaying the web page; and

the processor is further operative to instruct the communications device to transmit the request for updated data to the server in response to the timer reaching a predetermined value.

A METHOD AND SYSTEM FOR UPDATING OBJECTS CONTAINED WITHIN A WEB PAGE

ABSTRACT OF THE DISCLOSURE

A method and system for selectively updating selective objects of web page without requiring that the entire web page be refreshed, and without requiring user intervention. The invention takes the form of a web page containing an I-frame displayed in a web browser. An I-frame is a special type of frame with both a height and width of zero, and dimensioned so as to be effectively invisible through the use of programming commands. The I-frame may not be seen by a user of the present invention looking at a displayed web page. The web page also typically contains one or more updateable objects, and may contain one or more nonupdateable objects. Updateable objects may take the form of HTML objects, or may be any programmable object that capable of being refreshed. The I-frame code contains a timer. This timer runs until a predetermined value is met, at which point the I-frame requests updated data from a server. Typically, this data takes the form of instructions to the I-frame to alter the various updateable objects contained within the web page, but external to the I-frame. Thus, the I-frame may alter portions of the web page which it does not contain, and with which it otherwise has no direct interaction. By contrast, standard frames only update objects contained within those frames.

Attorney Docket No.:

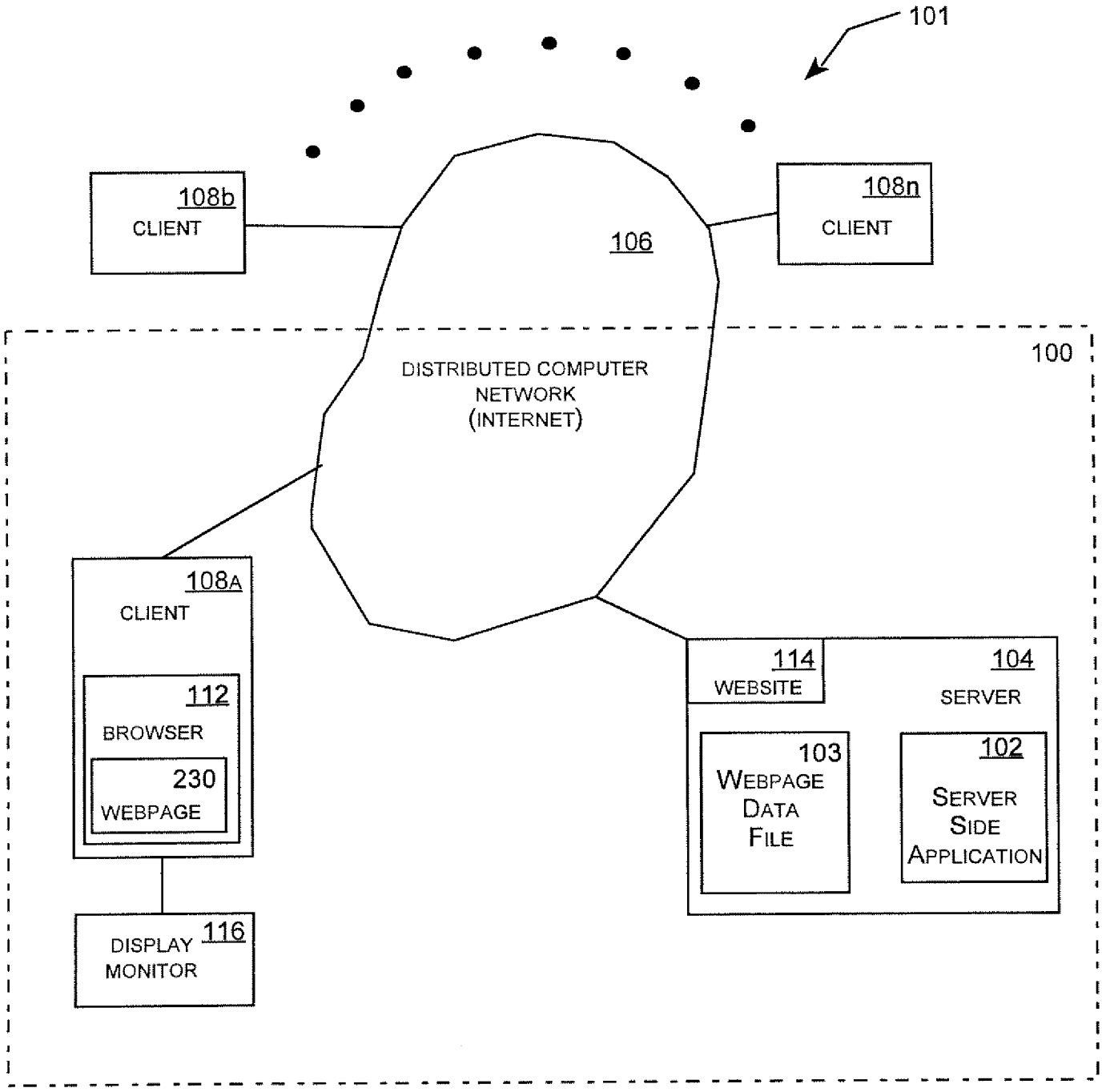
101
108b
CLIENT
106
108n
CLIENT
100
DISTRIBUTED COMPUTER
NETWORK
(INTERNET)
108A
CLIENT
112
BROWSER
230
WEBPAGE
DISPLAY
MONITOR
116
114
WEBSITE
104
SERVER
103
WEBPAGE
DATA
FILE
102
SERVER
SIDE
APPLICATION

FIG. 1

Fig. 2

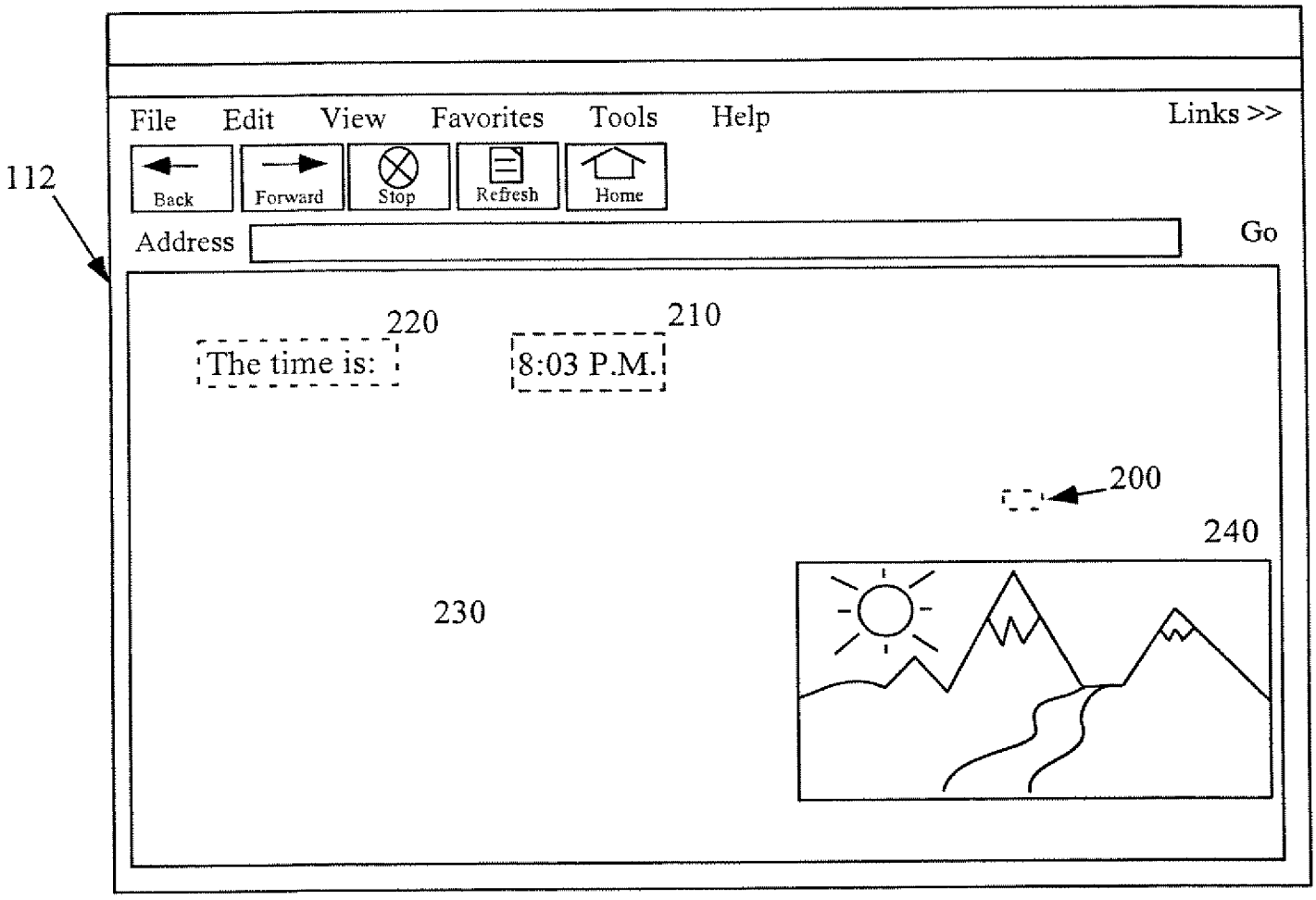
112
File
Edit
View
Favorites
Tools
Help
Links >>
Back
Forward
Stop
Refresh
Home
Address
Go
220
The time is:
210
8:03 P.M.
200
240
230

Fig. 3

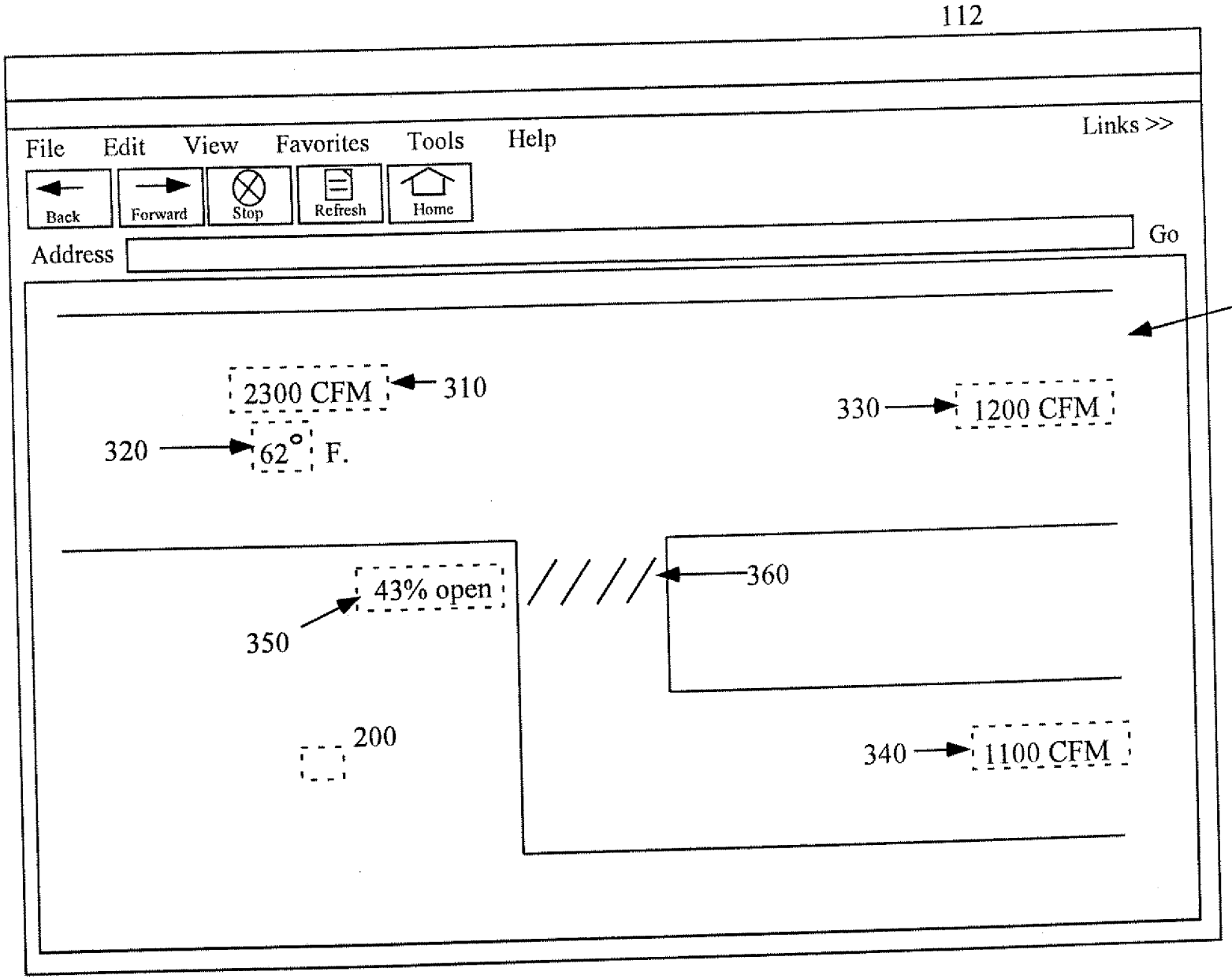
112
File
Edit
View
Favorites
Tools
Help
Links >>
Back
Forward
Stop
Refresh
Home
Address
Go
300
2300 CFM
310
320
62° F.
330
1200 CFM
43% open
350
360
200
340
1100 CFM

Fig. 4

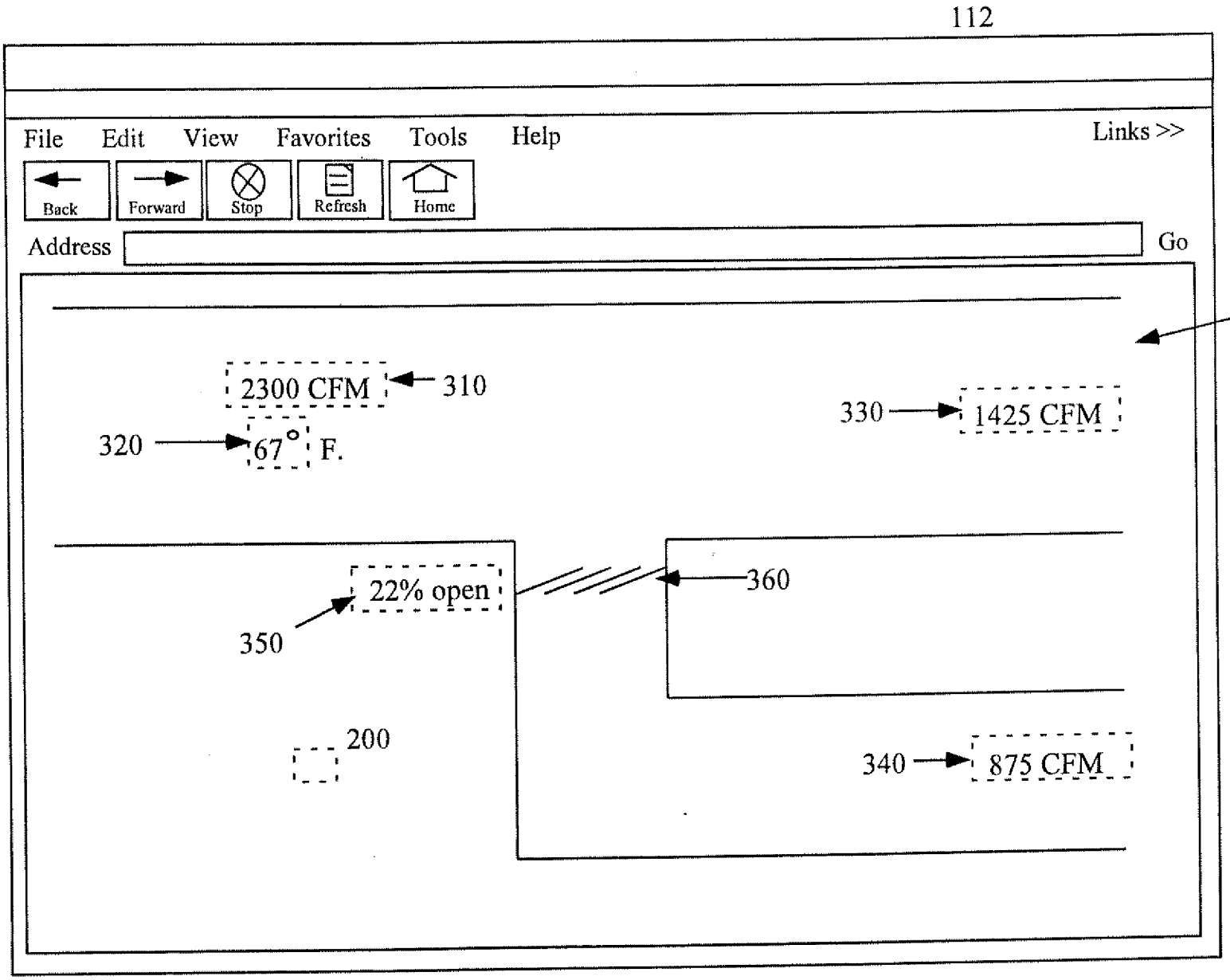
112
File
Edit
View
Favorites
Tools
Help
Links >>
Back
Forward
Stop
Refresh
Home
Address
Go
300
2300 CFM
310
320
67° F.
330
1425 CFM
22% open
350
360
200
340
875 CFM

Fig. 5

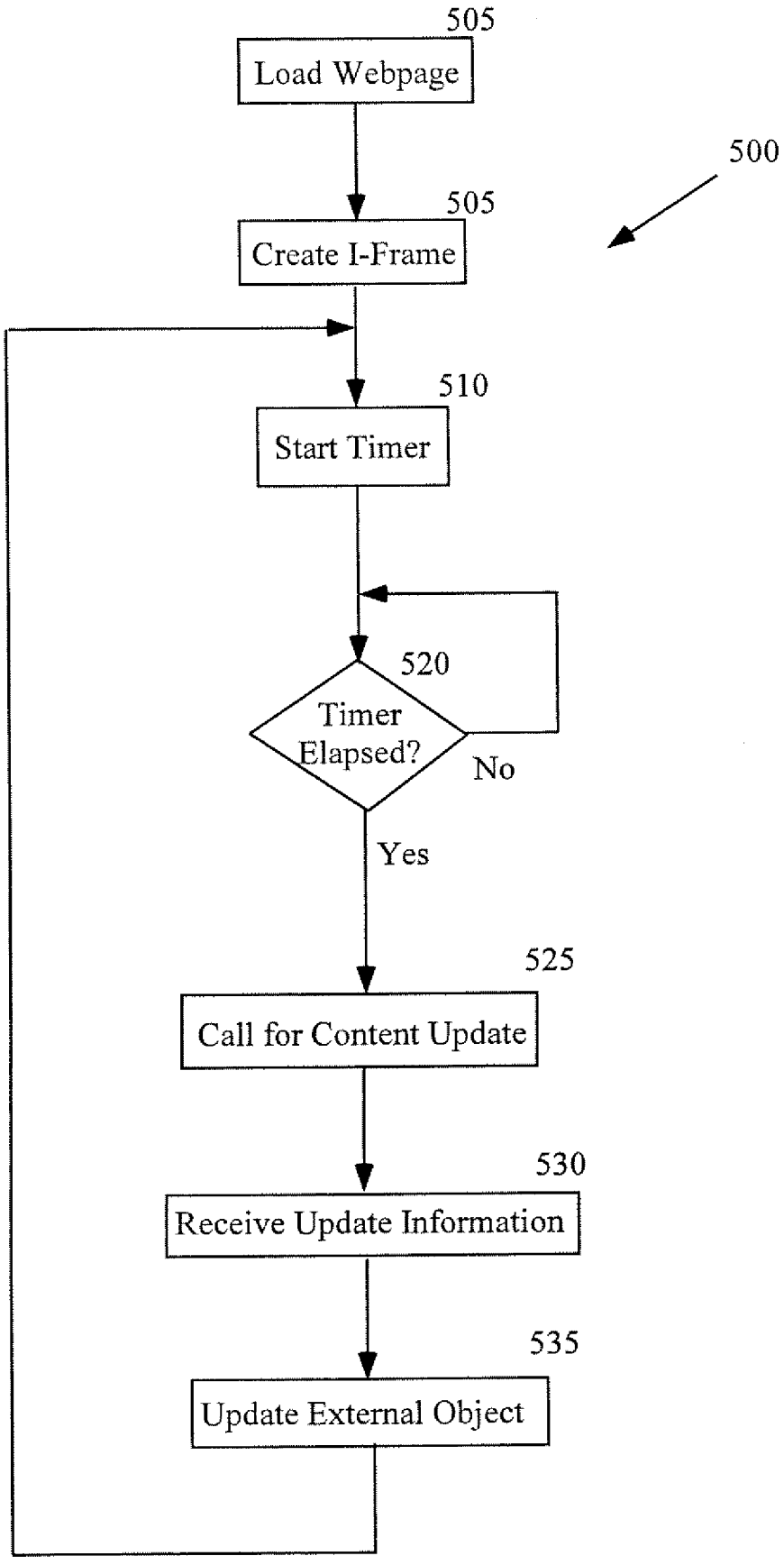
505
Load Webpage
500
505
Create I-Frame
510
Start Timer
520
Timer Elapsed?
No
Yes
525
Call for Content Update
530
Receive Update Information
535
Update External Object

EXHIBIT B

KILPATRICK STOCKTON LLP

November 10, 2000

Eric Craton
Vice President
Automated Logic Corporation
1150 Roberts Boulevard
Kennesaw, Georgia 30144

Re: Patent Applications entitled "A Method and System for Updating Objects Contained Within a Webpage" and

Our Files: 01360-0110;

Dear Eric:

As per our telephone conversation yesterday, I am enclosing a copy of the patent applications entitled "A Method and System for Updating Objects Contained Within a Webpage" and "
Both have been revised to incorporate all inventor comments received to date. We believe that each application is in final form, and is ready to be filed with the Patent Office upon your approval.

Additionally, we recommend completing a set of formal documents prior to filing these applications. The formal documents include an Inventor's Oath or Declaration and an Assignment conveying all rights from the inventor to Automated Logic Corporation. It is our understanding that Steve Appling is the sole inventor for both patents. A set of formal documents for each application is enclosed for Steve's review and signature. Once you approve the final drafts and return the executed documents, we will be pleased to file these applications on your behalf.

Please feel free to contact me at (404) 949-2497 if you have any questions regarding the applications.

Cordially yours,

KILPATRICK STOCKTON LLP

By: S. Craig Hemenway

SCH:ckf
Enclosures

From: Michael Pavento
To: sappling@automatedlogic.com
Date: Mon, Dec 11, 2000 8:05 PM
Subject: IFRAME

Steve,

Sorry I didn't send the revised IFRAME application to you last Friday as I had hoped. The application and its five figures are attached, all in WORD format. Also attached is a redline draft in WORD format. As with the last application, my changes were pretty significant and the redline may be more of a headache than its worth.

The formal papers for the IFRAME and applications are ready to be delivered to you as soon as you tell me the IFRAME application is ok. The formal papers are needed to complete the filing of the applications with the patent office.

The application doesn't seem like it needs too much work. I'll try to send that one to you for a last review ASAP.

-Mike Pavento
404.853.8422

Sutherland, Asbill & Brennan, LLP
999 Peachtree Street, NE
Atlanta, GA 30309
404.853.8000
Fax: 404.853.8806
www.sablaw.com

CC: ecraton@automatedlogic.com; Warren, Daniel J.

From: Michael Pavento
To: "sappling@automatedlogic.com".GWIA.SAB
Date: Fri, Dec 22, 2000 11:04 AM
Subject: Final IFRAME app.

Steve,
Attached is the final draft (I hope) of the IFRAME application. I revised the application per your comments and I added a few more claims at the end of the application.
A redline version is also attached so that you can see exactly what I've changed. I also attached the correct FIG. 1 so you can take a look at it.

Please call me as soon as you are finished looking over the final draft. I'd like to file the application today along with the application (we'll worry about getting your signature on formal documents later.)

If you have time, we could also discuss the application today. If not, we can wrap that one up next week (or when you return to the office from the Holidays.)

Mike Pavento
404.853.8422

Sutherland, Asbill & Brennan, LLP
999 Peachtree Street, NE
Atlanta, GA 30309
404.853.8000
Fax: 404.853.8806
www.sablaw.com

CC: ecraton@automatedlogic.com

From: <sappling@automatedlogic.com>
To: "Michael Pavento" <Mspavento@SABLAW.COM>
Date: Fri, Dec 22, 2000 12:36 PM
Subject: Re: Final IFRAME app.

Two comments - both in areas where you deleted something.

Bottom of page 6 - because you removed the line containing " ... a Java servlet is used..." the next sentence shouldn't contain "also" but should start "As is known in the art...".

Similar problem at the bottom of page 9 - since we removed the description of an embodiment using delta values we should also remove the word "Alternate" at the start of the last sentence.

Rest of it looks great.

Thanks for your help. Hope you have a great Holiday!